# #820 FactorialRec

Să se scrie o **funcție C++ recursivă** care returnează factorialul unui număr dat ca parametru.

int factorial(int n)

{if(n<=1) return 1;

return factorial(n-1)\*n; }

# #916 FactorialRec1

# Să se scrie o **funcție C++ recursivă** care determină factorialul unui număr transmis ca parametru și întoarce rezultatul prin intermediul unui parametru de ieșire.

int factorial (int n,int &f)

{ if (n<=1) f=1;

else

{ factorial (n-1,f);

f=f\*n;

} }

# #821 CmmdcRec

int cmmdc(int n, int m ) {

if (n%m==0) return m;

return cmmdc(m,n%m); }

# #917 CmmdcRec1

Să se scrie o **funcție C++ recursivă** care determină cel mai mare divizor comun a două numere transmise ca parametri și întoarce rezultatul prin intermediul unui parametru de ieșire.

void cmmdc (int a, int b, int &c ) {

if (a%b==0)

c=b;

else cmmdc(b,a%b,c); }

# #823 SumCifRec

Să se scrie o **funcție C++ recursivă** care să returneze suma cifrelor unui număr natural transmis ca parametru.

int sumcif (int a) {

if (a<10) return a;

return a%10+sumcif(a/10); }

# #918 SumCifRec1

Să se scrie o **funcție C++ recursivă** care determină suma cifrelor unui număr natural *n* transmis ca parametru și întoarce rezultatul prin intermediul unui parametru de ieșire.

void sumcif(int n, int &s)

{ if(n<=9)

s=n%10;

else

{ sumcif(n/10,s);

s+=n%10; } }

# #822 NrCifreZeroRec

Să se scrie o **funcție C++ recursivă** care să returneze numărul de cifre egale cu zero ale unui număr natural transmis ca parametru.

int nr\_cif\_zero(int n)

{

if(n<10)

{ if(n==0) return 1;

return 0; }

else return (n%10==0)+nr\_cif\_zero(n/10); }

# #919 NrCifreZeroRec1

Să se scrie o **funcție C++ recursivă** care să determine numărul de cifre egale cu zero ale unui număr natural transmis ca parametru și să întoarcă rezultatul prin intermediul unui parametru de ieșire.

void nr\_cif\_zero(int n,int &nr)

{ if(n<10) { if(n==0) nr=1; else nr=0; }

else {nr\_cif\_zero(n/10,nr); nr= (n%10==0)+nr; } }

# #824 CifMaxRec

**funcție C++ recursivă** care să returneze cifra maximă a unui număr natural

int cifmax(int n)

{

int c;

if(n<10) { return n; }

else {c=cifmax(n/10); if(c>n%10) return c; else return n%10; }

}

# #825 CifMinRec

Să se scrie o **funcție C++ recursivă** care să returneze cifra minimă a unui număr natural transmis ca parametru.

int cifmin(int n)

{int c;

if(n<10) { return n; }

else

{c=cifmin(n/10);

if(c<n%10) return c; else return n%10; }

}

# #920 CifMaxMinRec

Să se scrie o **funcție C++ recursivă** care să determine cifra maximă și cifra minimă a unui număr natural transmis ca parametru. Funcția va întoarce rezultatele prin intermediul unor parametri de ieșire.

void cifmaxmin(int n, int &maxi, int &mini)

{

if(n<10) { maxi=mini=n; }

else

{ cifmaxmin(n/10,maxi,mini);

if(mini>n%10) mini=n%10;

if(maxi<n%10) maxi=n%10;

} }

**#826** CifMinParRec

int cifminpar(int n) {

if(n<10) { if(n%2==0) return n; else return -1; }

else

{

int cif=n%10;

if(cif%2==0) {

if(cifminpar(n/10)!=-1) return min(cif,cifminpar(n/10));

else return cif;

}

else if(cifminpar(n/10)!=-1) { return cifminpar(n/10); }

return -1;

}

}

# #827 CifMaxParRec

int cifmaxpar(int n)

{ if(n<10) { if(n%2==0) return n;

return -1; }

else

{ if(n%10%2==0)

{

if(cifmaxpar(n/10)%2==0) return max(n%10,cifmaxpar(n/10));

return n%10;

}

if(cifmaxpar(n/10)%2==0)

{

return cifmaxpar(n/10);

}

return -1;

}

}

# #828 Manna-Pnueli

int mp(int n)

{

if(n>=12)

return n-1;

else

return mp(mp(n+2));

}

# #938 FSumRec

int suma(int a[],int n)

{

if(n==1)

return a[n-1];

return a[n-1]+suma(a,n-1);

}

# #829 AfisareRec

void afis()

{

int n;

cin>>n;

if(n==0)

cout<<0<<" ";

else

{

afis();

cout<<n<<" ";

}

}

# #836 AfisareVectorRec1

void afisvec(int a[],int n)

{

if(n==1)

cout<<a[n-1];

else

{

cout<<a[n-1]<<" ";

afisvec(a,n-1);

}

}

# #835 AfisareVectorRec

# Să se scrie o funcție C++ recursivă care afișează pe ecran elementele unui vector transmis ca parametru.

void afisvec(int a[],int n)

{

if(n>1)

afisvec(a,n-1);

cout<<a[n-1]<<" ";

}

# #924 MultipluRec

Scrieţi definiția completă a **subprogramului recursiv C++** *multiplu*care are *3* parametri: *a*, prin care primeşte un tablou unidimensional cu maximum *100* de numere naturale mai mici decât *1000*, *n*, numărul efectiv de elemente ale tabloului şi *k*, un număr natural.

Subprogramul returnează numărul de elemente din tablou care sunt multipli ai numărului *k* şi au ultima cifră egală cu *k*

int multiplu(int a[],int n,int k)

{

if(n==0)

return 0;

if(n>0)

{

if(a[n-1]%k==0&&a[n-1]%10==k)

return 1+multiplu(a,n-1,k);

else

return multiplu(a,n-1,k);

}

}